STRUCTURED WORLD REPRESENTATIONS VIA BLOCK-SLOT ATTENTION

Anonymous authors
Paper under double-blind review

ABSTRACT

In this paper, we propose a novel object-centric representation, called Block-Slot Representation which, unlike the conventional slot representation, provides concept-level disentanglement within a slot. A slot is constructed by composing a set of modular concept representations, called blocks, generated from a memory of concept prototypes. We call this slot construction process Block-Slot Attention. Block-Slot Attention facilitates the emergence of abstract concept blocks within a slot such as color, position, and texture, without any supervision. This brings the benefits of disentanglement into slots and the representation becomes more interpretable. Similar to Slot Attention, this mechanism can be used as a drop-in module in any arbitrary neural architecture. In experiments, we show that our model disentangles object properties significantly better than the previous methods, including complex textured scenes. We also demonstrate the ability to compose novel scenes by composing slots at the block-level.

1 INTRODUCTION

High-level cognitive abilities in humans such as reasoning, planning, analogy-making, and causal inference, seem to require the ability to manipulate knowledge pieces (McVee et al., 2005; Bottou, 2014; Schölkopf et al., 2021). Assuming that these knowledge pieces are somehow given, symbolic AI entertains good performance on some of these tasks (e.g., mathematics) where current deep learning may struggle. However, *how we can learn to obtain these knowledge pieces* is actually at the core of this challenge. Obtaining such knowledge pieces is particularly elusive in the case of images where, unlike text, a token-like structure is not naturally given.

Inspired by evidence from cognitive science (Spelke & Kinzler, 2007), object-centric representation learning (Greff et al., 2020b) tackles this problem by representing a scene compositionally with a set of constituent object representations, often called slots, via self-supervised learning. By learning to dynamically bind relevant visual signals into useful visual knowledge pieces — each of which has a distributed representation — it provides a way to learn symbol-like manipulation within the framework of deep representation learning.

Currently, one of the most popular approaches to object-centric representation learning is based on Slot Attention (SA) (Locatello et al., 2020; Singh et al., 2022a). In Slot Attention, each slot finds an input area to attend via competitive cross-attention between the slots and the input features. The slot representation is then updated by the input from the attended area, and this process repeats for a few iterations to refine the representation. In addition to the simplicity and good performance, being a deterministic mechanism, Slot Attention is also more generally applicable, as a drop-in module, to arbitrary neural architectures than previous approaches based on stochastic latent variable modeling (Greff et al., 2019; Burgess et al., 2019; Engelcke et al., 2019; Lin et al., 2020b).

One major limitation of Slot Attention is that, even though it does decompose a scene into a set of independent object representations (inter-slot disentanglement), it does not provide disentanglement *within* a slot representation (intra-slot disentanglement), i.e., for the factors of object-variation. This is contrasted in particular by the fact that the intra-slot disentanglement has naturally been achieved in the previous probabilistic approaches (Greff et al., 2019; Burgess et al., 2019; Engelcke et al., 2019) due to its use of the variational autoencoder framework (Kingma & Welling, 2013) that provides an explicit incentive for disentanglement. Considering various benefits of disentangled

representations (Higgins et al., 2017a), we therefore tackle the following question: "*Can we realize intra-slot disentanglement in Slot Attention?*"

In this paper, we propose a novel object-centric representation model to achieve this. In designing this, we pursue two main principles. The first is to make it as generic as possible by making it a simple deterministic layer like Slot Attention, as mentioned above. Second, while a factor in traditional approaches is a dimension of the representation vector, we seek to represent a factor of variation as a multi-dimensional vector. This has two benefits. First, being multi-dimensional, it provides a more flexible and richer way to represent a factor or property. Second, the vector representation is a more natural format to use as inputs to Transformers (Vaswani et al., 2017). This is of particular interest considering that the single-dimension representation of a factor is difficult to use as an input to a Transformer.

In this work, we realize these principles as follows. We propose a novel representation called the Block-Slot Representation or simply Block-Slots in which the representation of an object, i.e. a slot, is constructed by concatenating multi-dimensional vector modules called blocks. To learn this representation, we propose Block-Slot Attention (BSA), a learning mechanism that can encode a given input into Block-Slots. BSA works by performing attention on the input to find the local object areas and learns slots via iterative refinement. Crucially, the refinement pathways work independently for each block within a slot, leading to the decomposition of a slot into blocks. Adopting BSA as encoder in the image auto-encoding framework of Singh et al. (2022a), we find that BSA enables not just the emergence of objects in the slots like the conventional methods but also enables the emergence of blocks that represent within-slot factors of variation—all without supervision. In experiments, we show that our model disentangles object properties significantly better than the previous methods, including in complex textured scenes. Because BSA is simple and deterministic, it becomes possible to use it as a drop-in module in any arbitrary neural network model.

The contributions of the paper are as follows: (1) We propose the Block-Slot Attention (BSA) method, the first slot-attention-based representation learning mechanism which provides a **disentangled factor** structure within a slot. This is a general mechanism applicable to non-image domains as well. (2) This is also the first slot-based representation that can provide **vector-formed factor representations**, enabling the use of a transformer decoder for explicit factor-level relation modeling. (3) We show that using BSA, it is possible to develop and propose **a new object-centric model** for visual scene images that can disentangle the factors of object variation even for complex-textured scene images. This is also the **first demonstration of disentanglement within a slot on complex scenes** like the CLEVR-Tex. (4) We propose a **new method for binding block factors within a slot**. Finally, (5) we **empirically demonstrate the benefits and provide the ablations study** of the proposed model including performing novel compositional image generation.

2 Block-Slot Attention

In Block-Slot Representation, a slot is constructed by concatenating a set of vector modules called *blocks*. For images, each slot would represent one object in the scene, similar to the conventional slots. However, unlike the conventional slots which do not seek intra-slot disentanglement at all (Locatello et al., 2020) or seek to make each dimension an independent factor (Greff et al., 2019), we seek to make each block an independent factor. That is, each block is supposed to represent at most one property of the object and, conversely, each property is expected to be fully represented by a single block. Formally, by denoting the number of slots by N, the number of blocks per slot by M, and the dimension of a block by d, we denote the collection of N slots by a matrix $\mathbf{S} \in \mathbb{R}^{N \times Md}$. Each slot $\mathbf{s}_n \in \mathbb{R}^{Md}$ is a concatenation of M blocks. We use $\mathbf{s}_{n,m} \in \mathbb{R}^d$ to denote m-th block of slot n.

Block-Slot Attention (BSA) is a method we propose to encode a set of L input vectors into a set of slots $\mathbf{S}$. We note that this mechanism is generally applicable beyond images because it can, in principle, be used for any input (e.g., speech signals) as long as it is provided in the form of a set of vectors. Thus, it can be seen as a drop-in module that can be plugged in an arbitrary neural architecture. Our idea behind the design of BSA is two-fold. First, each slot $\mathbf{s}_n$ performs competitive attention on the input features to find a local area. Next, for each slot, the bottom-up information from the attended local area is split into M pathways—as many as the number of blocks. Here, each pathway works as an independent process that produces a block representation. We first initialize the

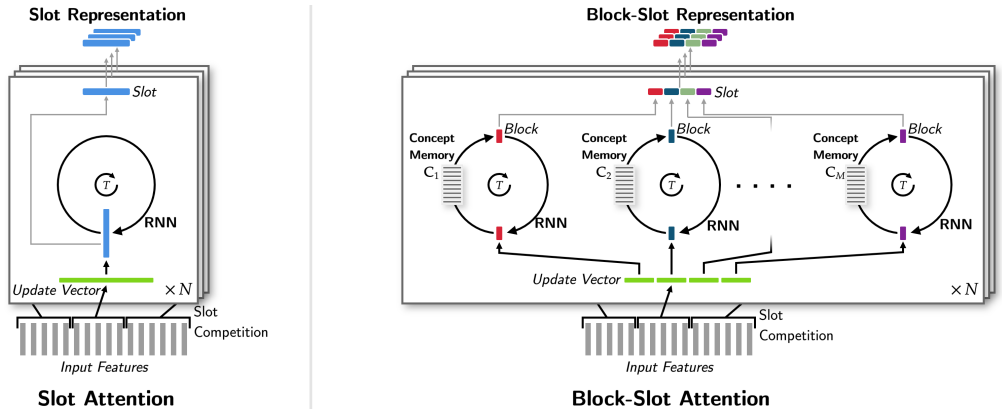

Figure 1: **Overview of Block-Slot Attention.** In Block-Slot Attention, the slots undergo a competition for each input feature, similar to Slot Attention. Different from Slot Attention, for each slot, the bottom-up information from the selected input features is split into several recurrent pathways, with each pathway producing a block representation. Crucially, each pathway contains a learned memory of concept prototypes shared across the m-th blocks in different slots. This is used to constrain the block representation to be a retrieved representation from the memory.

slots by randomly sampling them from a learned Gaussian distribution. We then refine these slots by repeating the above process multiple times.

Input Attention. In each refinement iteration, the slots $\mathbf{S}$ perform attention on the input features $\mathbf{E} \in \mathbb{R}^{L \times D}$. For this, the slots act as the queries, and the inputs act as the keys and the values. We apply linear projections q, k, and v on the slots and the inputs to generate the queries, keys, and values, all having the same dimension Md. We then perform dot-product between the queries and the keys to get the attention matrix $\mathbf{A} \in \mathbb{R}^{N \times L}$. We normalize the attention matrix by performing softmax along the axis N. This implements a competition among slots for attending over each input feature (Locatello et al., 2020). The attention matrix is then normalized along the axis L to make each row sum to 1. Finally, it is multiplied with the values $v(\mathbf{E}) \in \mathbb{R}^{L \times Md}$ to obtain the attention readout $\mathbf{U} \in \mathbb{R}^{N \times Md}$. In $\mathbf{U}$, the n-th row contains the attention result for slot n. We treat this row as a concatenation of M equal-sized vectors denoted by $\mathbf{u}_{n,1}, \ldots, \mathbf{u}_{n,M} \in \mathbb{R}^d$.

$$\mathbf{A} = \underset{N}{\texttt{softmax}}\left(\frac{q(\mathbf{S}) \cdot k(\mathbf{E})^T}{\sqrt{Md}}\right) \quad \Longrightarrow \quad \mathbf{A}_{n,l} = \frac{\mathbf{A}_{n,l}}{\sum_{l=1}^{L} \mathbf{A}_{n,l}} \quad \Longrightarrow \quad \mathbf{U} = \mathbf{A} \cdot v(\mathbf{E}),$$

Recurrent Block Refinement. We refine each block $\mathbf{s}_{n,m}$ using its corresponding readout $\mathbf{u}_{n,m}$. For this, we apply a $\texttt{GRU}_{\phi_m}$ and a residual $\texttt{MLP}_{\phi_m}$. In our design, we choose to maintain separate parameters ϕ_m for each m as this design can potentially scale better for complex large-scale problems where the number of blocks can be large. However, we note that depending on the context, it is also a sensible design to share the parameters $\phi_m = \phi$ for all m.

$$\mathbf{s}_{n,m} = \texttt{GRU}_{\phi_m}(\mathbf{s}_{n,m}, \mathbf{u}_{n,m}) \qquad \Longrightarrow \qquad \mathbf{s}_{n,m} \mathrel{+}= \texttt{MLP}_{\phi_m}(\texttt{LN}(\mathbf{s}_{n,m})).$$

Concept Memory Bottleneck. The next step is to make each block retrieve a representation from a concept memory. A concept memory $\mathbf{C}_m \in \mathbb{R}^{K \times d}$ is a set of K learnable vectors or K latent *prototype* vectors associated with the particular concept m. Importantly, the m-th blocks in each slot share the same concept memory $\mathbf{C}_m$. Each block $\mathbf{s}_{n,m}$ performs dot-product attention on its corresponding concept memory $\mathbf{C}_m$ and the attention result is used as the block representation.

$$\mathbf{s}_{n,m} = \left[\underset{K}{\texttt{softmax}}\left(\frac{\mathbf{s}_{n,m} \cdot \mathbf{C}_m^T}{\sqrt{d}}\right)\right] \cdot \mathbf{C}_m.$$

This step completes one refinement iteration. The above steps are repeated several times (3 times in our experiments). Then, the slots obtained from the last iteration are taken to be the final representation for providing it downstream. We provide a pseudo-code of our method in Algorithm 1.

3 OBJECT-CENTRIC LEARNING WITH BLOCK-SLOT ATTENTION

In this section, we propose an unsupervised object-centric learning method via Block-Slot Attention on images. For this, we adopt an auto-encoding approach in which we apply our proposed Block-Slot Attention to encode the image and then apply an image transformer decoder to reconstruct the image. Given an image $\mathbf{x} \in \mathbb{R}^{H \times W \times C}$, we first obtain a set of image features using a backbone encoder network, e.g., a CNN in our experiment. In this CNN, positional encodings are added in the second-last layer and the final output feature map is flattened to form the input set in the matrix form $\mathbf{E} \in \mathbb{R}^{L \times D}$. We then encode $\mathbf{E}$ via Block-Slot Attention and obtain a set $\mathbf{S}$ of slots.

3.1 BLOCK BINDING

One crucial benefit realized by the Block-Slot representation is that the relationships and interactions among the representations can be modeled explicitly at the level of blocks rather than the level of slots. To obtain this benefit, we would like to provide a set of block vectors $\{\mathbf{s}_{n,m}\}$ as the input tokens to the transformer decoder. However, because a set is order-less, it is difficult for the decoder (1) to know which concept each block is representing and (2) to identify to which slot each block is belonging. Following the traditional approach for dealing with the order issue in Transformer, one possible approach to this problem is to provide two positional embeddings to each block indexed by (n, m), one for slot membership n and the other for concept id m. However, this would mean that the decoder should take the burden of learning invariance to the slot order which is not built-in into the design. Furthermore, it could be difficult for the decoder to generalize well when many more slots than the number seen during training are given at test time.

To avoid these challenges, we propose the following method. To each block, we add a positional encoding $\mathbf{p}_m^{\text{block}}$ that depends only on m but not on n. We then let the blocks belonging to the same slot interact via a 1-layer transformer, termed *block binding*, as follows:

$$\bar{\mathbf{s}}_{n,m} = \mathbf{s}_{n,m} + \mathbf{p}_m^{\text{block}} \qquad \Longrightarrow \qquad \tilde{\mathbf{s}}_{n,1}, \ldots, \tilde{\mathbf{s}}_{n,M} = \text{BlockBinding}(\bar{\mathbf{s}}_{n,1}, \ldots, \bar{\mathbf{s}}_{n,M})$$

After this, we provide all the returned vectors to the transformer decoder as a single set $\mathcal{S} = \{\tilde{\mathbf{s}}_{n,m}\}$. With this approach, the decoder is informed about the role of each block via the positional encoding $\mathbf{p}_m^{\text{block}}$. Because the blocks in the same slot have interacted with each other, the information about which blocks belong to the same slot is also implicitly available to the decoder. Furthermore, the decoder treats the slots as order-invariant and can also accept more slots than shown during training.

3.2 AUTOREGRESSIVE DECODING

Conditioned on $\mathcal{S}$ obtained after block binding, the transformer decoder learns to decode the image. Similar to the decoder of SLATE (Singh et al., 2022a) and STEVE (Singh et al., 2022b), the transformer does not directly reconstruct the pixels of the image. Instead, conditioned on $\mathcal{S}$, the transformer auto-regressively reconstructs a discrete token representation $\mathbf{z}_1, \ldots, \mathbf{z}_{L'}$ of the given image $\mathbf{x}$ provided by a discrete VAE. To predict a token $\mathbf{z}_l$ at position l using the transformer, we take the previous tokens $\mathbf{z}_1, \ldots, \mathbf{z}_{l-1}$ of the sequence as input and retrieve a learned embedding for each token. We then add positional encodings to these. The resulting embeddings $\mathbf{e}_1, \ldots, \mathbf{e}_{l-1}$ are given as input to the transformer decoder to predict the next token:

$$\mathbf{e}_l = \text{Dictionary}_\theta(\mathbf{z}_l) + \mathbf{p}_l^{\text{token}} \qquad \Longrightarrow \qquad \mathbf{o}_l = \text{TransformerDecoder}_\theta(\mathbf{e}_1, \ldots, \mathbf{e}_{l-1}; \mathcal{S}),$$

where $\mathbf{o}_l$ are the predicted logits for token at position l. The training is done by learning to predict all tokens in parallel using causal masking via the learning objective $\mathcal{L} = \sum_{l=1}^{L'} \text{CrossEntropy}(\mathbf{z}_l, \mathbf{o}_l)$. See Appendix D.4 for more details.

4 EVALUATING BLOCK-SLOT REPRESENTATION

In this section, we introduce the method to quantitatively evaluate Block-Slot representations based on the DCI framework (Eastwood & Williams, 2018). The DCI framework is multi-faceted and evaluates multiple aspects of the quality of a representation vector in terms of Disentanglement (D), Completeness (C), and Informativeness (I). Although the original DCI framework is designed for a

single vector representation, Nanbo et al. (2020) proposed a method to make it applicable to a set of slots by matching the slots with the true objects using the IoU score.

Specifically, let $\mathcal{S} = (\mathbf{s}_1, \ldots, \mathbf{s}_O)$ be a set of slots, with a slot $\mathbf{s}_o \in \mathbb{R}^d$. Also, assume that for each object, there are I independent ground-truth factors of object variation such as color, shape, and position. We then train K probe functions $g_1^{\text{probe}}, \ldots, g_I^{\text{probe}}$, one for each factor, in such a way so as to predict the factor label y_o^i from slot vector $\mathbf{s}_o$ i.e. $g_i^{\text{probe}} : \mathbb{R}^d \to \mathbb{N}$. The probes are classifiers implemented using gradient-boosted trees Locatello et al. (2019). Using the trained probes, we can obtain a feature importance matrix $\mathbf{R} = (R_{i,j}) \in \mathbb{R}^{I \times d}$. Each entry $R_{i,j}$ denotes the importance of feature dimension j in predicting the factor i and is obtained from the corresponding probe g_i^{probe}. Using the importance matrix, the disentanglement-score $\texttt{D}_j$ for a specific feature dimension j and the completeness-score $\texttt{C}_i$ for a specific factor i are computed as follows:

$$\texttt{D}_j = 1 - H_I(R_{:,j}), \qquad \texttt{C}_i = 1 - H_d(R_{i,:}),$$

where $H_I(\cdot)$ denotes entropy with log-base I and $H_d(\cdot)$ denotes entropy with log-base d. From these, the final disentanglement-score $\texttt{D}$ and completeness-score $\texttt{C}$ are computed as the weighted average of $\texttt{D}_j$ over all the dimensions $j = 1, \ldots, d$ and similarly for $\texttt{C}_i$ by averaging over $i = 1, \ldots, I$. The informativeness-score is the average of the prediction accuracy of all the probes.

For evaluating Block-Slots, we first apply the same probing approach as described above to obtain the importance matrix $\mathbf{R} = (R_{i,j}) \in \mathbb{R}^{I \times Md}$, where M is the number of blocks, d is the block size, and Md is the size of the slot. To evaluate the representation in the setting where each block is an independent factor, we obtain an importance score per block by summing the importance scores across the dimensions belonging to the same block, giving us block importance matrix $\mathbf{R}^{\text{block}} = (R_{i,m}^{\text{block}}) \in \mathbb{R}^{I \times M}$. Using these block importance values, we compute the disentanglement-scores and the completeness-scores in the same way as described above.

5 Related Work

Our work belongs in the line of unsupervised object-centric learning methods (Greff et al., 2016; Burgess et al., 2019; Greff et al., 2019; Locatello et al., 2020; Greff et al., 2017; Engelcke et al., 2020; 2021; Anciukevicius et al., 2020; von Kügelgen et al., 2020; Greff et al., 2020a; Singh et al., 2022a; Chang et al., 2022). In this line, intra-slot disentanglement has been explored inside the VAE framework (Greff et al., 2019; Burgess et al., 2019; Nanbo et al., 2020; Zoran et al., 2021). However, unlike these, our work achieves this goal without taking the complexity of probabilistic modeling. Another parallel line of work pursues disentanglement of object properties by adopting a spatial transformer decoder – a special decoder designed to accept explicit values of object size and position (Jaderberg et al., 2015; Eslami et al., 2016; Crawford & Pineau, 2019b; Lin et al., 2020b; Jiang & Ahn, 2020; Deng et al., 2020; Chen et al., 2021). However, these cannot scale to other factors of variation such as color or material. Prabhudesai et al. (2021) seek object and property-level disentanglement but require the use of 3D viewpoint, occupancy labels, and an auxiliary consistency loss. Learning neural networks composed of independent competing modules has been explored by Goyal et al. (2021b; 2020); Lamb et al. (2021); Goyal et al. (2021a). Our prototype learning is related to van den Oord et al. (2017); Liu et al. (2021); Caron et al. (2020) and our concept memory attention is related to (Ramsauer et al., 2021). However, these works do not focus on object-centric learning or intra-slot disentanglement.

Disentangling single-vector scene representations has been pursued in Chen et al. (2016); Higgins et al. (2017); Kim & Mnih (2018b); Kumar et al. (2017); Chen et al. (2018); Locatello et al. (2019); Montero et al. (2021). Parallel to these, Hu et al. (2022); Zhou et al. (2020) pursue block-level disentanglement but on 3D mesh data and in single object scenes. Stammer et al. (2021) use prototypes to learn block-wise disentangled codes. However, this is not unsupervised and does not deal with multiple objects, unlike ours. Several metrics have also been proposed alongside the above works to measure disentanglement in single vector representations (Higgins et al., 2017b; Kim & Mnih, 2018a; Chen et al., 2018; Eastwood & Williams, 2018; Ridgeway & Mozer, 2018; Kumar et al., 2018; Andreas, 2019). For slot representations, some newly proposed metrics exist but these do not apply to block-level disentanglement within slots (Racah & Chandar, 2020; Xie et al., 2022; Nanbo et al., 2020). The metric of Dang-Nhu (2022) applies to block-level disentanglement. However, it involves simultaneously optimizing slot permutation and property prediction which could be unstable. Also, it requires building a much larger feature-importance matrix which can be extremely slow.

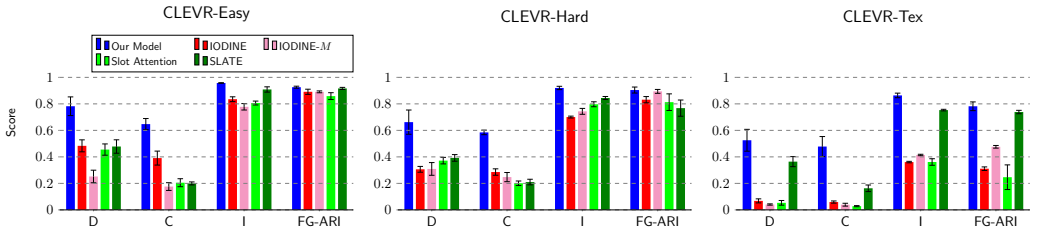

Figure 2: **Slot Learning and Intra-Slot Disentanglement.** We compare our model with the baselines in terms of Disentanglement (D), Completeness (C), Informativeness (I), and FG-ARI.

6 Experiments

Datasets. We evaluate our model on three datasets: CLEVR-Easy, CLEVR-Hard, and CLEVR-Tex. These are variants of the original CLEVR dataset (Johnson et al., 2017) having large object sizes to make object properties such as shape clearly visible in the image. These datasets have different difficulty levels: *i)* In CLEVR-Easy, there are only three ground-truth factors of variation, i.e., color, shape, and position. These are varied while fixing the material and the size. As in the original CLEVR, the color takes values from a small palette of 8 colors. *ii)* In CLEVR-Hard, all 5 properties i.e. color, shape, position, material, and size are the factors of variation. Furthermore, the color takes values from a much larger palette containing 137 colors. *iii)* The CLEVR-Tex is derived from the original CLEVR-Tex dataset (Karazija et al., 2021). Here, the objects and the backgrounds can take materials from a total of 57 possible materials, much more complex than the previous two datasets. In addition to the material, the object shapes and positions are varied. Our model only sees the raw images as input and no other supervision is provided. For evaluation, we use the ground-truth object masks and property factors. These labels are available as the datasets are synthetically generated. See Fig. 5 for sample images of these datasets.

Baselines. We compare our model with three state-of-the-art unsupervised object-centric representation models: IODINE (Greff et al., 2019), Slot Attention (Locatello et al., 2020), and SLATE (Singh et al., 2022a). IODINE represents the class of VAE-based object-centric models while Slot Attention and SLATE are two deterministic object-centric models. The number of independent factors per slot for the baseline models is their slot size. Following their original papers, we set this to 64. In our model, the number of independent factor modules per slot is the number of blocks M. We use 8, 16, and 8 blocks for CLEVR-Easy, CLEVR-Hard, and CLEVR-Tex, respectively. Importantly, having such few factor modules, e.g., 8 or 16, is a desirable property of our model because it leads to better interpretability within a slot. As such, we test how IODINE would perform with such few factor modules. We test this by training IODINE by matching its slot size to the number of blocks M of our model (termed as IODINE-M). Because using such small slot sizes deviates from IODINE's original hyperparameters, we also analyze a wide range of other slot sizes for IODINE in Fig. 6 and find that it does not lead to much improvement in IODINE's performance.

6.1 Object and Property Disentanglement

To compare how well the models decompose the image into slots and then the slots into the factors of variation, we use four metrics. To evaluate object segmentation, we adopt the Adjusted Rand Index applied to the foreground objects (FG-ARI), as in prior works (Greff et al., 2019; Locatello et al., 2020). FG-ARI has also been shown to correlate with downstream task performance (Dittadi et al., 2022). To evaluate intra-slot disentanglement, we use DCI as introduced in Section 4.

Quantitative Results. We show a quantitative evaluation of object-level and property-level disentanglement in Fig. 2. We note that our model significantly outperforms the baselines in all 3 datasets, nearly doubling the disentanglement and completeness-scores in CLEVR-Easy and CLEVR-Hard. In CLEVR-Tex, the other models except for SLATE completely fail in handling the visual complexity of the textured scenes while our model remains robust. To the best of our knowledge, it is for the first time in this line of research that property-level disentanglement has been achieved at this level of

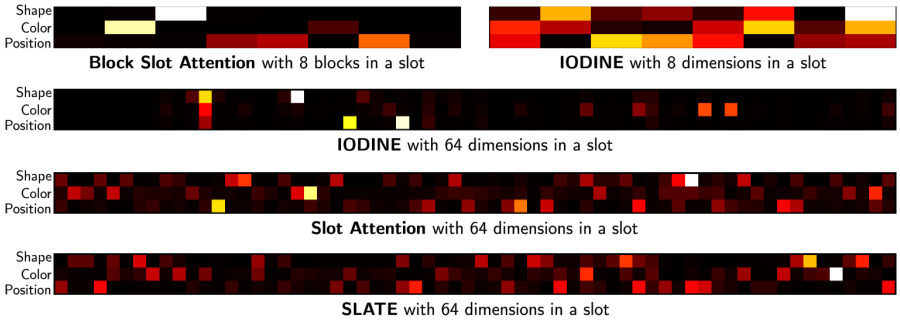

Figure 3: **Feature Importances in CLEVR-Easy.** We visualize the intra-slot feature importance matrices of our model and the baselines. Each row corresponds to a ground-truth object property. Each column corresponds to a block for our model and to a dimension for the other models.

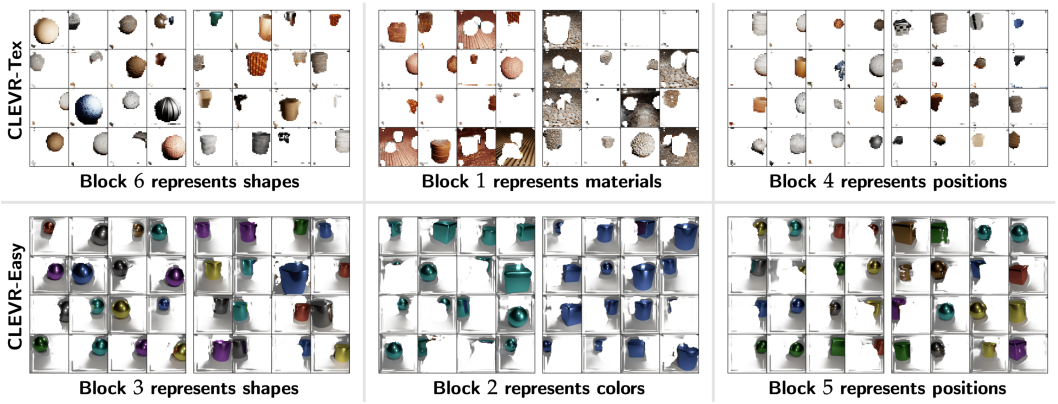

Figure 4: **Visualization of object clusters obtained by applying k-means on specific blocks in CLEVR-Tex and CLEVR-Easy.** We observe that blocks specialize to properties such as shape, color, material, and position and abstract away the other properties. For object clusters in the CLEVR-Hard dataset, see Figure 10 in Appendix C.

textural complexity. The informativeness-score and FG-ARI also suggest that our model is superior in property prediction and object segmentation to the other methods.

In Fig. 3, we visualize the feature importance matrices used for computing the DCI scores in CLEVR-Easy. We note that our importance matrix is significantly more sparse than other methods. We also note that deterministic models like SLATE and Slot Attention have significantly more active dimensions than VAE-based IODINE due to the regularizing effect of the VAE prior. This observation also correlates with lower completeness-scores of Slot Attention and SLATE relative to IODINE. Such effect of the VAE prior was also noted in Dang-Nhu (2022).

Qualitative Results. In Fig. 4, we visualize the learned representation space of each block. For this, we feed a large batch of images to our model, collect the m-th blocks, and then apply k-means clustering on the collected block representations. By visualizing the objects belonging to a cluster of block m, we observe the emergence of semantically meaningful abstract concepts in blocks. For CLEVR-Easy as an example, in $m = 3$ we can see shape concepts like sphere or cylinder; in $m = 2$ color concepts like blue and green; and also the position concepts in $m = 5$. It is remarkable that this emergence also occurs in the visually complex CLEVR-Tex dataset—for the first time in this line of research. It is somewhat surprising to see all the spheres of CLEVR-Tex dataset are clustered together despite the fact that their surface appearances and textures are very different from each other.

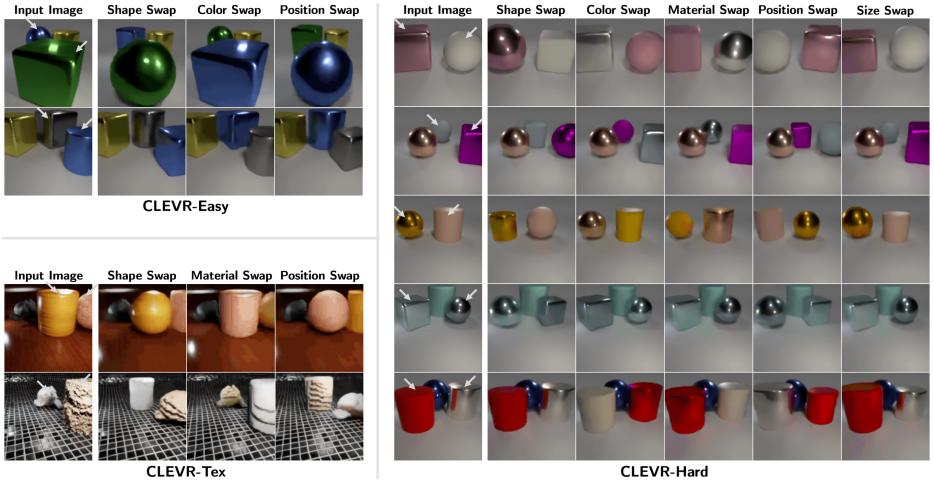

Figure 5: **Visualization of Property-level Scene Manipulation.** We manipulate a given scene by swapping a specific property of two objects in the scene. For a given scene, we choose two objects and swap their shape, color, position, material, and size. White arrows shown on the input images point to two objects in each scene whose properties are swapped. For more samples, see Figures 11, 12, and 13 in Appendix C.

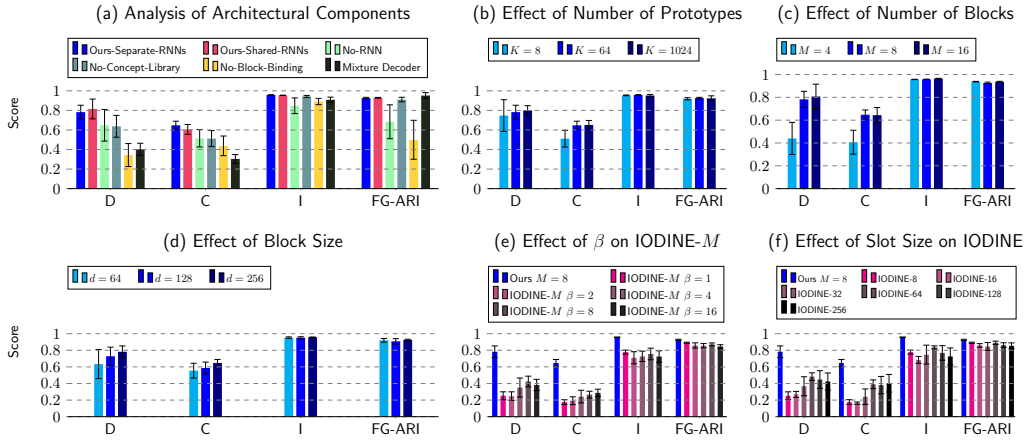

Figure 6: **Analysis.** We evaluate the role of various design choices and hyperparameters for our model and the baseline IODINE. In all plots, we report the DCI and the FG-ARI.

6.2 Property-level Scene Composition

In Fig. 5, we show that our model can generate novel images by swapping the properties of objects in a given scene. We do this by first representing a given image as slots using our model. We then choose two slots and select the blocks which capture a specific object property such as color. We then swap these selected blocks between the two slots. These are then provided to the decoder to generate the scene. We note that our model executes the property swap effectively without affecting other properties. This shows that our blocks are modular. We also found the blocks to be interpretable and compact which enabled us to easily identify which blocks to swap in order to swap a specific property. Furthermore, our model demonstrates this ability not only in simple scenes but also in complex textured scenes of CLEVR-Tex. It is also noteworthy that our generated scenes show accurate object reflections even after the properties are swapped. This is due to the Transformer decoder that is able to model interactions between slots when generating the pixels.

6.3 ANALYSIS

In this section, we analyze several variants and hyperparameters for the evaluated models. We perform these evaluations on the CLEVR-Easy dataset and report the results in Fig. 6.

Role of Architectural Components of Block-Slot Attention. In Fig. 6 (a), we analyze the role of various architectural components of our model. First, we compare the effect of sharing the parameters of the RNN used during block update (i.e. the GRU and the MLP) relative to having separate RNN parameters for each block. We find no significant difference in performance. Nevertheless, having a separate RNN per block may serve as a more flexible and modular design for more complex large-scale problems. Second, we test the performance by removing the RNN used during the block refinement. We find that this has a significant adverse effect on the segmentation quality as well as the DCI performance, showing that the RNN is a crucial component. Third, we test the role of having the concept memory bottleneck. We find that while this does not affect the segmentation and informativeness of the slots, however, it significantly reduces the disentanglement and the completeness-scores. Fourth, we test the role of the block binding mechanism used for interfacing the Block-Slot representation with the transformer decoder. We test this by training a variant of our model in which we provide all the blocks as a single set to the decoder. We find that performance drops severely with some seeds failing completely. This is because without knowing the role of each block and which blocks are bound together, the decoder cannot learn to decode the image well.

Role of Transformer Decoder. In Fig. 6 (a), we also analyze the role of the transformer decoder in our model and whether using the mixture-based decoder, a common choice in several object-centric learning methods, can also provide a similar performance or not. For this, we adopt the decoder design of Locatello et al. (2020). We decode each object component by concatenating the blocks of a slot and providing it to a spatial broadcast decoder Watters et al. (2019). Interestingly, we find that this leads to a large drop in the disentanglement and the completeness-score, suggesting that the role of the transformer decoder is crucial.

Effect of Number of Prototypes, Number of Blocks, and Block Size. In Fig. 6 (b), (c), and (d), we study the effect of the number of prototypes K, the number of blocks M, and block size d on the DCI performance. First, with the increasing number of prototypes, we note improvement, especially in the completeness-score. This suggests that more prototypes increase the expressiveness of each block. Because of this, fewer blocks are needed to represent a single factor which improves the completeness score. Second, for the number of blocks $M = 4$, we see low disentanglement and completeness scores. Because this dataset has 3 factors of object variation, this suggests that 4 blocks may be too few for them to train well. However, going to $M = 8$ and $M = 16$, the performance improves and then saturates. Third, with increasing block sizes, we note a steady increase in performance.

Analysis of IODINE. In Fig. 6 (e) and (f), we analyze IODINE. We test the effect of adopting the β-VAE objective (Higgins et al., 2017a) and the effect of varying the slot size. While larger β leads to a slight increase in the disentanglement-score and the completeness-score, it still remains significantly worse than ours. Large β also leads to a slight drop in the informativeness-score due to the greater regularizing effect of the KL term. Increasing the slot size d which is equivalent to increasing the number of independent factor modules in IODINE slightly improves performance but our model is still much better. We also note that while we test IODINE with slot size as large as 256, having few factor modules, e.g., 8 or 16, is more desirable since it leads to better interpretability within a slot.

7 CONCLUSION

In this study, we propose a new structured representation learning mechanism, Block-Slot Attention (BSA). Unlike traditional Slot Attention, BSA learns to disentangle the factors of variation within a slot. Also, BSA provides the factor representations in a vector-form, called block, rather than a single dimensional variable. Using BSA, we also introduce a new object-centric representation learning method for scene images. To achieve this, we also found that our new method for binding the blocks belonging to the same slot is crucial in achieving good performance. The experiment results demonstrate various benefits of the disentanglement produced by the proposed model. In Appendix F, we discuss our limitations and the future works.

REPRODUCIBILITY STATEMENT

We will release the code and the datasets used in this work at the link: `https://github.com/blockslotattention`.

ETHICS STATEMENT

Future extensions and applications arising from our work should be mindful of the environmental impact of training large-scale models. They should actively avoid its potential misuse in surveillance or by generating synthetic images with malicious intent. However, it is unlikely that the model in its current form would lead to such an impact in the near future. Our model also has the potential for making a positive impact in areas such as scene understanding, robotics, and autonomous navigation.

REFERENCES

Titas Anciukevicius, Christoph H Lampert, and Paul Henderson. Object-centric image generation with factored depths, locations, and appearances. *arXiv preprint arXiv:2004.00642*, 2020.

Jacob Andreas. Measuring compositionality in representation learning. *ArXiv*, abs/1902.07181, 2019.

Federico Baldassarre and Hossein Azizpour. Towards self-supervised learning of global and object-centric representations. *ArXiv*, abs/2203.05997, 2022.

Beril Besbinar and Pascal Frossard. Self-supervision by prediction for object discovery in videos. *arXiv preprint arXiv:2103.05669*, 2021.

Léon Bottou. From machine learning to machine reasoning. *Machine learning*, 94(2):133–149, 2014.

Christopher P Burgess, Loic Matthey, Nicholas Watters, Rishabh Kabra, Irina Higgins, Matt Botvinick, and Alexander Lerchner. Monet: Unsupervised scene decomposition and representation. *arXiv preprint arXiv:1901.11390*, 2019.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *ArXiv*, abs/2006.09882, 2020.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV 2021 - International Conference on Computer Vision*, 2021.

Michael Chang, Thomas L Griffiths, and Sergey Levine. Object representations as fixed points: Training iterative refinement algorithms with implicit differentiation. *arXiv preprint arXiv:2207.00787*, 2022.

Chang Chen, Fei Deng, and Sungjin Ahn. Roots: Object-centric representation and rendering of 3d scenes. *Journal of Machine Learning Research*, 22(259):1–36, 2021.

Tian Qi Chen, Xuechen Li, Roger B Grosse, and David K Duvenaud. Isolating sources of disentanglement in variational autoencoders. In *Advances in Neural Information Processing Systems*, pp. 2610–2620, 2018.

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in neural information processing systems*, pp. 2172–2180, 2016.

Eric Crawford and Joelle Pineau. Exploiting spatial invariance for scalable unsupervised object tracking. *arXiv preprint arXiv:1911.09033*, 2019a.

Eric Crawford and Joelle Pineau. Spatially invariant unsupervised object detection with convolutional neural networks. In *Proceedings of AAAI*, 2019b.

Eric Crawford and Joelle Pineau. Learning 3d object-oriented world models from unlabeled videos. In *Workshop on Object-Oriented Learning at ICML*, 2020.

Antonia Creswell, Kyriacos Nikiforou, Oriol Vinyals, Andre Saraiva, Rishabh Kabra, Loic Matthey, Chris Burgess, Malcolm Reynolds, Richard Tanburn, Marta Garnelo, et al. Alignnet: Unsupervised entity alignment. *arXiv preprint arXiv:2007.08973*, 2020.

Antonia Creswell, Rishabh Kabra, Chris Burgess, and Murray Shanahan. Unsupervised object-based transition models for 3d partially observable environments. *arXiv preprint arXiv:2103.04693*, 2021.

Raphaël Dang-Nhu. Evaluating disentanglement of structured representations. In *ICLR*, 2022.

Fei Deng, Zhuo Zhi, Donghun Lee, and Sungjin Ahn. Generative scene graph networks. In *International Conference on Learning Representations*, 2020.

Andrea Dittadi, Samuele Papa, Michele De Vita, Bernhard Schölkopf, Ole Winther, and Francesco Locatello. Generalization and robustness implications in object-centric learning. In *ICML*, 2022.

Yilun Du, Kevin Smith, Tomer Ulman, Joshua Tenenbaum, and Jiajun Wu. Unsupervised discovery of 3d physical objects from video. *arXiv preprint arXiv:2007.12348*, 2020.

Yilun Du, Shuang Li, Yash Sharma, Josh Tenenbaum, and Igor Mordatch. Unsupervised learning of compositional energy concepts. *Advances in Neural Information Processing Systems*, 34, 2021a.

Yilun Du, Kevin A. Smith, Tomer Ulman, Joshua B. Tenenbaum, and Jiajun Wu. Unsupervised discovery of 3d physical objects from video. *ArXiv*, abs/2007.12348, 2021b.

Cian Eastwood and Christopher K. I. Williams. A framework for the quantitative evaluation of disentangled representations. In *ICLR*, 2018.

Gamaleldin F. Elsayed, Aravindh Mahendran, Sjoerd van Steenkiste, Klaus Greff, Michael Curtis Mozer, and Thomas Kipf. Savi++: Towards end-to-end object-centric learning from real-world videos. *ArXiv*, abs/2206.07764, 2022.

Martin Engelcke, Adam R. Kosiorek, Oiwi Parker Jones, and Ingmar Posner. Genesis: Generative scene inference and sampling with object-centric latent representations, 2019.

Martin Engelcke, Adam R. Kosiorek, Oiwi Parker Jones, and Ingmar Posner. Genesis: Generative scene inference and sampling with object-centric latent representations. In *ICLR 2020 : Eighth International Conference on Learning Representations*, 2020.

Martin Engelcke, Oiwi Parker Jones, and Ingmar Posner. Genesis-v2: Inferring unordered object representations without iterative refinement. *arXiv preprint arXiv:2104.09958*, 2021.

SM Ali Eslami, Nicolas Heess, Theophane Weber, Yuval Tassa, David Szepesvari, and Geoffrey E Hinton. Attend, infer, repeat: Fast scene understanding with generative models. In *Advances in Neural Information Processing Systems*, pp. 3225–3233, 2016.

Anirudh Goyal, Alex Lamb, Phanideep Gampa, Philippe Beaudoin, Sergey Levine, Charles Blundell, Yoshua Bengio, and Michael C. Mozer. Object files and schemata: Factorizing declarative and procedural knowledge in dynamical systems. *ArXiv*, abs/2006.16225, 2020.

Anirudh Goyal, Aniket Didolkar, Nan Rosemary Ke, Charles Blundell, Philippe Beaudoin, Nicolas Manfred Otto Heess, Michael C. Mozer, and Yoshua Bengio. Neural production systems. *ArXiv*, abs/2103.01937, 2021a.

Anirudh Goyal, Alex Lamb, Jordan Hoffmann, Shagun Sodhani, Sergey Levine, Yoshua Bengio, and Bernhard Schölkopf. Recurrent independent mechanisms. *ArXiv*, abs/1909.10893, 2021b.

Klaus Greff, Antti Rasmus, Mathias Berglund, Tele Hao, Harri Valpola, and Jürgen Schmidhuber. Tagger: Deep unsupervised perceptual grouping. In *Advances in Neural Information Processing Systems*, pp. 4484–4492, 2016.

Klaus Greff, Sjoerd van Steenkiste, and Jürgen Schmidhuber. Neural expectation maximization. In *Advances in Neural Information Processing Systems*, pp. 6691–6701, 2017.

Klaus Greff, Raphaël Lopez Kaufmann, Rishab Kabra, Nick Watters, Chris Burgess, Daniel Zoran, Loic Matthey, Matthew Botvinick, and Alexander Lerchner. Multi-object representation learning with iterative variational inference. *arXiv preprint arXiv:1903.00450*, 2019.

Klaus Greff, Sjoerd van Steenkiste, and Jürgen Schmidhuber. On the binding problem in artificial neural networks. *arXiv preprint arXiv:2012.05208*, 2020a.

Klaus Greff, Sjoerd van Steenkiste, and Jürgen Schmidhuber. On the binding problem in artificial neural networks. *arXiv preprint arXiv:2012.05208*, 2020b.

Zhen He, Jian Li, Daxue Liu, Hangen He, and David Barber. Tracking by animation: Unsupervised learning of multi-object attentive trackers. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1318–1327, 2019.

Olivier J. H'enaff, Skanda Koppula, Evan Shelhamer, Daniel Zoran, Andrew Jaegle, Andrew Zisserman, João Carreira, and Relja Arandjelovi'c. Object discovery and representation networks. *ArXiv*, abs/2203.08777, 2022.

Paul Henderson and Christoph H. Lampert. Unsupervised object-centric video generation and decomposition in 3d. *ArXiv*, abs/2007.06705, 2020.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *ICLR 2017 : International Conference on Learning Representations 2017*, 2017.

Irina Higgins, Loïc Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew M Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *ICLR*, 2017a.

Irina Higgins, Loïc Matthey, Arka Pal, Christopher P. Burgess, Xavier Glorot, Matthew M. Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *ICLR*, 2017b.

Xue Hu, Xinghui Li, Benjamin Busam, Yiren Zhou, Ale Leonardis, and Shanxin Yuan. Disentangling 3d attributes from a single 2d image: Human pose, shape and garment. *ArXiv*, abs/2208.03167, 2022.

Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. In *Advances in neural information processing systems*, pp. 2017–2025, 2015.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

Jindong Jiang and Sungjin Ahn. Generative neurosymbolic machines. *arXiv preprint arXiv:2010.12152*, 2020.

Jindong Jiang, Sepehr Janghorbani, Gerard de Melo, and Sungjin Ahn. SCALOR: generative world models with scalable object representations. In *International Conference on Learning Representations*, 2020.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2901–2910, 2017.

Rishabh Kabra, Daniel Zoran, Goker Erdogan, Loic Matthey, Antonia Creswell, Matthew Botvinick, Alexander Lerchner, and Christopher P Burgess. Simone: View-invariant, temporally-abstracted object representations via unsupervised video decomposition. *arXiv preprint arXiv:2106.03849*, 2021.

Laurynas Karazija, Iro Laina, and C. Rupprecht. Clevrtex: A texture-rich benchmark for unsupervised multi-object segmentation. *ArXiv*, abs/2111.10265, 2021.

Hyunjik Kim and Andriy Mnih. Disentangling by factorising. *ArXiv*, abs/1802.05983, 2018a.

Hyunjik Kim and Andriy Mnih. Disentangling by factorising. *arXiv preprint arXiv:1802.05983*, 2018b.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Thomas Kipf, Gamaleldin F. Elsayed, Aravindh Mahendran, Austin Stone, Sara Sabour, Georg Heigold, Rico Jonschkowski, Alexey Dosovitskiy, and Klaus Greff. Conditional Object-Centric Learning from Video. *arXiv preprint arXiv:2111.12594*, 2021.

Adam Kosiorek, Hyunjik Kim, Yee Whye Teh, and Ingmar Posner. Sequential attend, infer, repeat: Generative modelling of moving objects. In *Advances in Neural Information Processing Systems*, pp. 8606–8616, 2018.

Abhishek Kumar, Prasanna Sattigeri, and Avinash Balakrishnan. Variational inference of disentangled latent concepts from unlabeled observations. In *International Conference on Learning Representations*, 2017.

Abhishek Kumar, Prasanna Sattigeri, and Avinash Balakrishnan. Variational inference of disentangled latent concepts from unlabeled observations. *ArXiv*, abs/1711.00848, 2018.

Alex Lamb, Di He, Anirudh Goyal, Guolin Ke, Chien-Feng Liao, Mirco Ravanelli, and Yoshua Bengio. Transformers with competitive ensembles of independent mechanisms. In *arXiv e-prints*, 2021.

Zhixuan Lin, Yi-Fu Wu, Skand Vishwanath Peri, Jindong Jiang, and Sungjin Ahn. Improving generative imagination in object-centric world models. In *International Conference on Machine Learning*, pp. 4114–4124, 2020a.

Zhixuan Lin, Yi-Fu Wu, Skand Vishwanath Peri, Weihao Sun, Gautam Singh, Fei Deng, Jindong Jiang, and Sungjin Ahn. Space: Unsupervised object-oriented scene representation via spatial attention and decomposition. In *International Conference on Learning Representations*, 2020b.

Dianbo Liu, Alex Lamb, Kenji Kawaguchi, Anirudh Goyal, Chen Sun, Michael Curtis Mozer, and Yoshua Bengio. Discrete-valued neural communication. In *NeurIPS*, 2021.

Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Raetsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *International Conference on Machine Learning*, pp. 4114–4124, 2019.

Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention, 2020.

Sindy Löwe, Klaus Greff, Rico Jonschkowski, Alexey Dosovitskiy, and Thomas Kipf. Learning object-centric video models by contrasting sets. *arXiv preprint arXiv:2011.10287*, 2020.

Sindy Lowe, Phillip Lippe, Maja R. Rudolph, and Max Welling. Complex-valued autoencoders for object discovery. *ArXiv*, abs/2204.02075, 2022.

Mary B McVee, Kailonnie Dunsmore, and James R Gavelek. Schema theory revisited. *Review of educational research*, 75(4):531–566, 2005.

Milton Llera Montero, Casimir JH Ludwig, Rui Ponte Costa, Gaurav Malhotra, and Jeffrey Bowers. The role of disentanglement in generalisation. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=qbH974jKUVy`.

Li Nanbo, Cian Eastwood, and Robert B Fisher. Learning object-centric representations of multi-object scenes from multiple views. In *Advances in Neural Information Processing Systems*, 2020.

Mihir Prabhudesai, Shamit Lal, Darshan Patil, Hsiao-Yu Fish Tung, Adam W. Harley, and Katerina Fragkiadaki. Disentangling 3d prototypical networks for few-shot concept learning. *ArXiv*, abs/2011.03367, 2021.

Evan Racah and A. P. Sarath Chandar. Slot contrastive networks: A contrastive approach for representing objects. *ArXiv*, abs/2007.09294, 2020.

Hubert Ramsauer, Bernhard Schafl, Johannes Lehner, Philipp Seidl, Michael Widrich, Lukas Gruber, Markus Holzleitner, Milena Pavlovi'c, Geir Kjetil Sandve, Victor Greiff, David P. Kreil, Michael Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. Hopfield networks is all you need. *ArXiv*, abs/2008.02217, 2021.

Karl Ridgeway and Michael C. Mozer. Learning deep disentangled embeddings with the f-statistic loss. *ArXiv*, abs/1802.05312, 2018.

Mehdi S. M. Sajjadi, Daniel Duckworth, Aravindh Mahendran, Sjoerd van Steenkiste, Filip Pavetić, Mario Lučić, Leonidas J. Guibas, Klaus Greff, and Thomas Kipf. Object Scene Representation Transformer. *NeurIPS*, 2022.

Bernhard Schölkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. Toward causal representation learning. *Proceedings of the IEEE*, 109(5):612–634, 2021.

Gautam Singh, Skand Peri, Junghyun Kim, Hyunseok Kim, and Sungjin Ahn. Structured world belief for reinforcement learning in pomdp. In *International Conference on Machine Learning*, pp. 9744–9755. PMLR, 2021.

Gautam Singh, Fei Deng, and Sungjin Ahn. Illiterate dall-e learns to compose. In *International Conference on Learning Representations*, 2022a. URL `https://openreview.net/forum?id=h0OYV0We3oh`.

Gautam Singh, Yi-Fu Wu, and Sungjin Ahn. Simple unsupervised object-centric learning for complex and naturalistic videos. *ArXiv*, abs/2205.14065, 2022b.

Elizabeth S Spelke and Katherine D Kinzler. Core knowledge. *Developmental science*, 10(1):89–96, 2007.

Wolfgang Stammer, Marius Memmel, Patrick Schramowski, and Kristian Kersting. Interactive disentanglement: Learning concepts by interacting with their prototype representations. *ArXiv*, abs/2112.02290, 2021.

Aleksandar Stanić and Jürgen Schmidhuber. R-sqair: relational sequential attend, infer, repeat. *arXiv preprint arXiv:1910.05231*, 2019.

Karl Stelzner, Kristian Kersting, and Adam R Kosiorek. Decomposing 3d scenes into objects via unsupervised volume segmentation. *arXiv preprint arXiv:2104.01148*, 2021.

Aaron van den Oord, Oriol Vinyals, and koray kavukcuoglu. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, volume 30, pp. 6306–6315, 2017.

Sjoerd Van Steenkiste, Michael Chang, Klaus Greff, and Jürgen Schmidhuber. Relational neural expectation maximization: Unsupervised discovery of objects and their interactions. *arXiv preprint arXiv:1802.10353*, 2018.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Rishi Veerapaneni, John D Co-Reyes, Michael Chang, Michael Janner, Chelsea Finn, Jiajun Wu, Joshua B Tenenbaum, and Sergey Levine. Entity abstraction in visual model-based reinforcement learning. *arXiv preprint arXiv:1910.12827*, 2019.

Julius von Kügelgen, Ivan Ustyuzhaninov, Peter Gehler, Matthias Bethge, and Bernhard Schölkopf. Towards causal generative scene models via competition of experts. *arXiv preprint arXiv:2004.12906*, 2020.

Yangtao Wang, Xiaoke Shen, Shell Xu Hu, Yuan Yuan, James L. Crowley, and Dominique Vaufreydaz. Self-supervised transformers for unsupervised object discovery using normalized cut. *ArXiv*, abs/2202.11539, 2022.

Nicholas Watters, Loic Matthey, Matko Bosnjak, Christopher P Burgess, and Alexander Lerchner. Cobra: Data-efficient model-based rl through unsupervised object discovery and curiosity-driven exploration. *arXiv preprint arXiv:1905.09275*, 2019.

Nicholas Watters, Loic Matthey, Christopher P. Burgess, and Alexander Lerchner. Spatial broadcast decoder: A simple architecture for learning disentangled representations in vaes. *arXiv preprint arXiv:1901.07017*, 2019.

Marissa A Weis, Kashyap Chitta, Yash Sharma, Wieland Brendel, Matthias Bethge, Andreas Geiger, and Alexander S Ecker. Unmasking the inductive biases of unsupervised object representations for video sequences. *arXiv preprint arXiv:2006.07034*, 2, 2020.

Xin Wen, Bingchen Zhao, Anlin Zheng, X. Zhang, and Xiaojuan Qi. Self-supervised visual representation learning with semantic grouping. *ArXiv*, abs/2205.15288, 2022.

Yi-Fu Wu, Jaesik Yoon, and Sungjin Ahn. Generative video transformer: Can objects be the words? In *International Conference on Machine Learning*, pp. 11307–11318. PMLR, 2021.

Sirui Xie, Ari S. Morcos, Song-Chun Zhu, and Ramakrishna Vedantam. Coat: Measuring object compositionality in emergent representations. In *ICML*, 2022.

Peiyu Yu, Sirui Xie, Xiaojian Ma, Yixin Zhu, Ying Nian Wu, and Song-Chun Zhu. Unsupervised foreground extraction via deep region competition. In *NeurIPS*, 2021.

Keyang Zhou, Bharat Lal Bhatnagar, and Gerard Pons-Moll. Unsupervised shape and pose disentanglement for 3d meshes. In *ECCV*, 2020.

Daniel Zoran, Rishabh Kabra, Alexander Lerchner, and Danilo J Rezende. Parts: Unsupervised segmentation with slots, attention and independence maximization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10439–10447, 2021.

A BLOCK-SLOT ATTENTION

We provide the pseudo-code of the proposed Block-Slot Attention approach below.

Algorithm 1 Block-Slot Attention. The algorithm receives the input features $\mathbf{E} \in \mathbb{R}^{L \times D}$; the number of slots N; the number of blocks M; and the block size d. The model parameters include: the linear projections $k,\ q,\ v$ with output dimension Md; GRU and MLP networks for each m i.e. $\texttt{GRU}_{\phi_1} \ldots \texttt{GRU}_{\phi_M}$ and $\texttt{MLP}_{\phi_1} \ldots \texttt{MLP}_{\phi_M}$; the learned concept memories $\mathbf{C}_1, \ldots, \mathbf{C}_M \in \mathbb{R}^{K \times d}$; and a Gaussian distribution's mean and diagonal covariance $\boldsymbol{\mu}, \boldsymbol{\sigma} \in \mathbb{R}^{Md}$.

```
S = Tensor(N, Md)
S ~ N(μ, σ)
E = LayerNorm(E)
for t = 1...T sequentially:
        S = LayerNorm(S)
        A = Softmax(1/√Md q(S) · k(E)^T, axis='slots')
        A = A / A.Sum(axis='inputs', keepdim=True)
        U = A · v(E)
        for n = 1...N and m = 1...M in parallel:
                s_{n,m} = GRU_{φ_m}(state=s_{n,m}, update=u_{n,m})
                s_{n,m} += MLP_{φ_m}(LayerNorm(s_{n,m}))
                s_{n,m} = Softmax(1/√d s_{n,m} · C_m^T, axis='prototypes') · C_m
return S
```

B ADDITIONAL RELATED WORK

Unsupervised Object-Centric Learning. Object-centric learning has also been explored extensively in dynamic scenes (Kosiorek et al., 2018; Stanić & Schmidhuber, 2019; Jiang et al., 2020; Crawford & Pineau, 2019a; Lin et al., 2020a; Wu et al., 2021; Singh et al., 2021; He et al., 2019; Greff et al., 2017; Van Steenkiste et al., 2018; Veerapaneni et al., 2019; Watters et al., 2019; Weis et al., 2020; Du et al., 2020; Kipf et al., 2021; Kabra et al., 2021; Zoran et al., 2021; Besbinar & Frossard, 2021; Creswell et al., 2020; 2021; Elsayed et al., 2022); and in 3D scenes (Chen et al., 2021; Henderson & Lampert, 2020; Crawford & Pineau, 2020; Stelzner et al., 2021; Du et al., 2021b; Kabra et al., 2021; Sajjadi et al., 2022; Nanbo et al., 2020; Singh et al., 2022b). However, adopting Block-Slot representation in dynamic and 3D scenes is a future direction and is orthogonal to our current focus. Aside from these auto-encoding-based methods, alternative frameworks have also shown promise, notably compositional energy-based scene modeling (Du et al., 2021a; Yu et al., 2021), complex-valued neural networks Lowe et al. (2022), and reconstruction-free representation learning (Caron et al., 2021; Löwe et al., 2020; Wang et al., 2022; H'enaff et al., 2022; Wen et al., 2022; Baldassarre & Azizpour, 2022). However, these methods do not disentangle object properties.

C ADDITIONAL RESULTS

C.1 EXTENDED ANALYSIS OF OUR MODEL

In this section, we perform additional ablations of our model that shed further light on it.

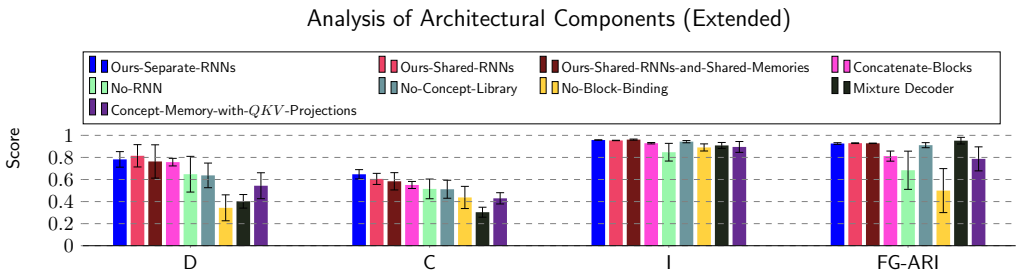

Figure 7: **Extended Analysis of Our Model.** We report Disentanglement (D), Completeness (C), Informativeness (I), and FG-ARI.

In Fig. 7, we show all the ablation results together, including those that we could not include in the main paper due to a limitation of space. Below, we analyze these ablation results.

Sharing the concept-memory across blocks. We test what would be the effect of sharing not just the RNNs across the blocks but also sharing the concept memory across the blocks. We can see that as the level of weight-sharing increases, the 'completeness' score decreases. In other words, with more weight-sharing among blocks, the model tends to use more blocks on average to represent a single factor. However, as this is not a drastic degradation, we can say that weight-separation is not the major factor responsible for the emerged disentanglement. What is crucial for getting good disentanglement is doing each block's prototype attention and refinement independently.

Concatenating blocks to construct a slot instead of Block-Binding. In this ablation, we test an alternative to the block-binding approach: *what if we simply concatenate the blocks in a slot and directly provide the set of slots as input to the transformer decoder?* We find that this variant achieves a significantly worse performance in the FG-ARI score while also achieving a slightly worse performance in the DCI scores.

Linear projections in concept-memory attention. In this ablation, we test the effect of applying linear projections on the query, key, and value when performing attention on the concept memory as follows:

$$\mathbf{s}_{n,m} = \left[\underset{K}{\texttt{softmax}} \left(\frac{q_m(\mathbf{s}_{n,m}) \cdot k_m(\mathbf{C}_m)^T}{\sqrt{d}} \right) \right] \cdot v_m(\mathbf{C}_m).$$

Here, q_m, k_m, and v_m are the per-block linear projections for computing the queries, the keys, and the values. We find that this variant of our model is much more unstable and performs significantly worse than our default architecture.

C.2 ANALYSIS OF CONCEPT MEMORY AND PROTOTYPES

In this section, we qualitatively analyze several questions regarding the concept-memory bottleneck and the prototypes.

Does concept memory attention perform hard-selection of a single prototype? We analyze the attention weights of a block on the concept memory in Fig. 8. We note that the weights are not one-hot and thus the model is not making a single hard selection of prototype. Rather, it is performing a soft selection of multiple activated prototypes. Thus, its capacity is not limited to the K prototype vectors but can be much more flexible.

How is the coverage of prototype usage? We analyze the average activation of each prototype across a large batch of input images in the CLEVR-Easy dataset. We show these average activations in Fig. 9. We find that a large majority of prototypes seem to be active.

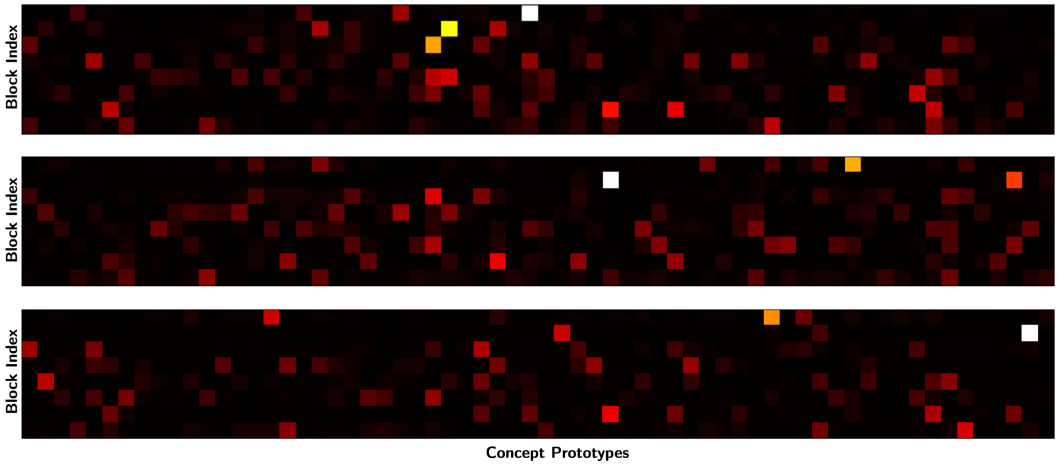

Figure 8: **Prototype Attention Weights.** Three instances of blocks attending to the prototypes in the concept memory. A cell (m, k) in the visualized matrices corresponds to a block $m = 1, \ldots, 8$ and a prototype vector $k = 1, \ldots, 64$ from the concept memory of the corresponding block. We note that the attention patterns are not one-hot selection of a single prototype. Instead, it is a soft and flexible linear combination of multiple prototypes.

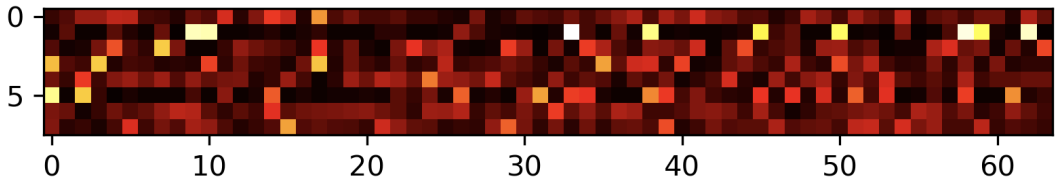

Figure 9: **Prototype Coverage.** We show the attention weight of each prototype averaged over a batch of 300 input images of CLEVR-Easy dataset. Each cell (m, k) in the visualized matrix corresponds to a block $m = 1, \ldots, 8$ and a prototype $k = 1, \ldots, 64$ from its corresponding concept memory $\mathbf{C}_m$.

C.3 Computational Requirements

In Table 1, we compare the memory requirements of our model with respect to IODINE, our main baseline that pursues intra-slot disentanglement. Here, we can see that our memory consumption is better than that of IODINE. We also note that our training speed is slightly slower than that of IODINE. However, we must interpret these results with care and take into account the fact that IODINE performs significantly worse than our model when it comes to the disentanglement performance.

	IODINE	Our Model
Memory Consumption	37.30 GiB	**31.01 GiB**
Time per Training Iteration	**0.84s**	1.01s

Table 1: **Comparison of computational requirements between IODINE and our model.** For fair comparison, all values are measured on training runs with the same batch size of 40.

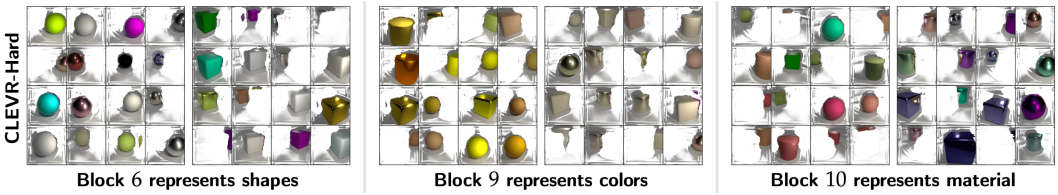

Figure 10: Object clusters obtained by applying k-means on specific blocks in CLEVR-Hard. We observe that blocks specialize to properties such as shape, color and material and abstract away the other properties.

Figure 11: Additional samples of property-swapping in CLEVR-Easy.

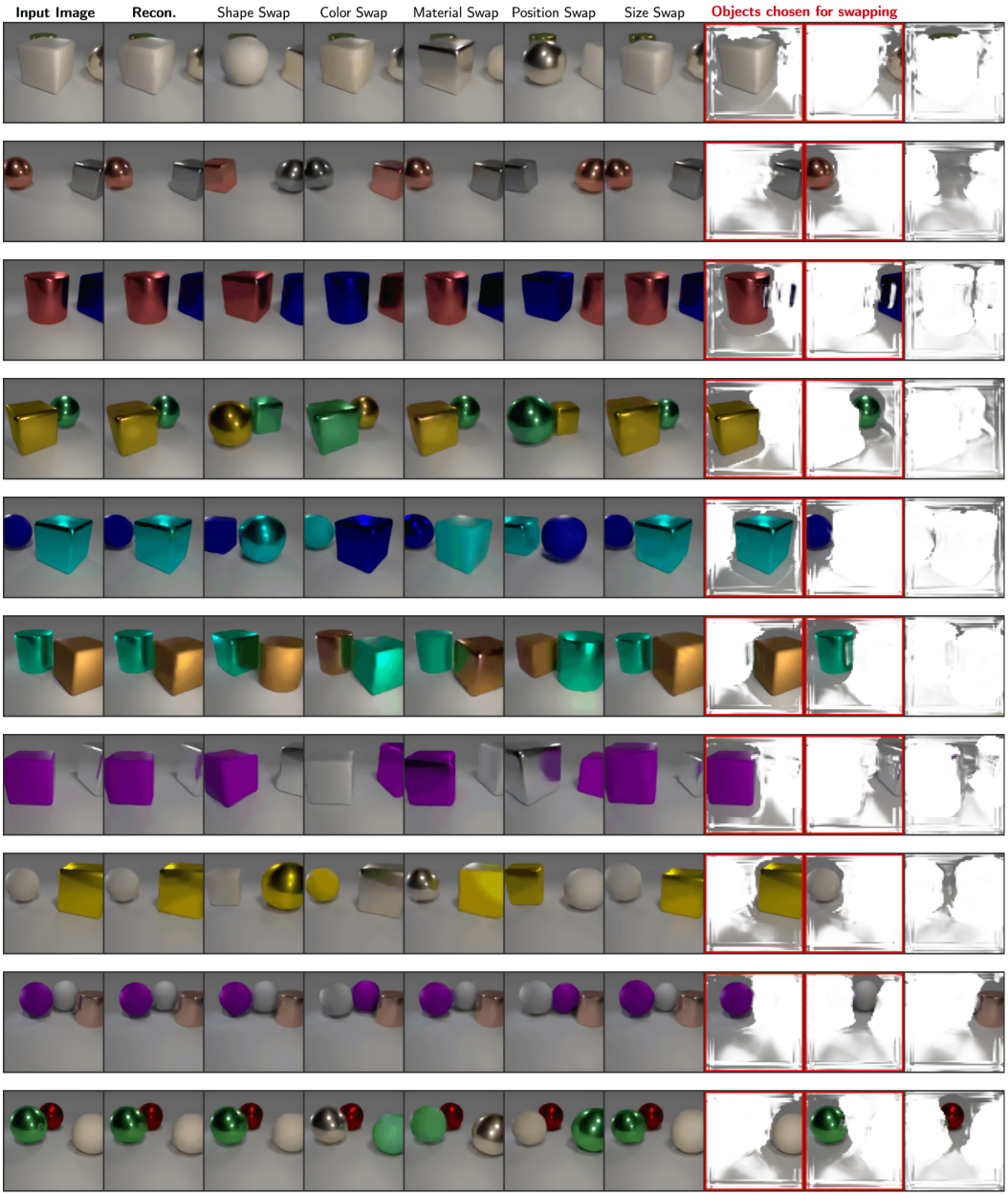

Figure 12: Additional samples of property-swapping in CLEVR-Hard.

Figure 13: Additional samples of property-swapping in CLEVR-Tex.

D ADDITIONAL IMPLEMENTATION DETAILS

Module	Hyperparameter	Dataset		
		CLEVR-Easy	CLEVR-Hard	CLEVR-Tex
General	Batch Size	40	40	40
	Training Steps	200K	200K	400K
Block-Slot Attention	Block Size	256	128	256
	# Blocks	8	16	8
	# Prototypes	64	64	64
	# Iterations	3	3	3
	# Slots	4	4	6
	Learning Rate	0.0001	0.0001	0.0001
Transformer Decoder	# Decoder Blocks	8	8	8
	# Decoder Heads	4	4	8
	Hidden Size	192	192	192
	Dropout	0.1	0.1	0.1
	Learning Rate	0.0003	0.0003	0.0003
dVAE	Learning Rate	0.0003	0.0003	0.0003
	Patch Size	4×4 pixels	4×4 pixels	4×4 pixels
	Vocabulary Size	4096	4096	4096
	Temperature Start	1.0	1.0	1.0
	Temperature End	0.1	0.1	0.1
	Temperature Decay Steps	30000	30000	30000

Table 2: Hyperparameters of our model used in our experiments.

D.1 BACKBONE IMAGE ENCODER

The backbone image encoder consists of a CNN encoder that encodes the image into a feature map. The specifications of this CNN encoder are described in Table 3. For CLEVR-Tex dataset, we found it useful to use a deeper CNN network described in Table 4. After applying the CNN, positional encodings are added to the feature map. Layer-normalization is applied on the resulting feature map followed by a 2-layer MLP with hidden dimension 192. The resulting feature map is flattened along the spatial dimensions to produce a set of input features that are then provided to the Block-Slot Attention module. The positional encodings are learned in a similar way as in Slot Attention by mapping a 4-dimensional grid of position coordinates to a 192-dimensional vector via a linear projection.

Layer	Kernel Size	Stride	Padding	Channels	Activation
Conv	5×5	2	2	512	ReLU
Conv	5×5	1	2	512	ReLU
Conv	5×5	1	2	512	ReLU
Conv	5×5	1	2	192	None

Table 3: Specifications of CNN layers of the backbone image encoder in CLEVR-Easy and CLEVR-Hard datasets.

D.2 RECURRENT BLOCK UPDATE

The block update is performed using a GRU network and a 2-layer MLP network with dimensions equal to the block size. For maintaining the separation of GRU and MLP model parameters between blocks and to apply GRU and MLP on all the M blocks in parallel, we adopt the implementation of Goyal et al. (2021b).

Layer	Kernel Size	Stride	Padding	Channels	Activation
Conv	5×5	2	2	512	ReLU
Conv	5×5	1	2	512	ReLU
Conv	5×5	2	2	512	ReLU
Conv	5×5	1	2	512	ReLU
Conv	5×5	1	2	192	None

Table 4: Specifications of CNN layers of the backbone image encoder in CLEVR-Tex dataset.

D.3 Concept Memory

To facilitate the training of the concept memories $\mathbf{C}_1, \ldots, \mathbf{C}_M$, we parameterize prototypes as an MLP projection of a learned vector: $\mathbf{C}_{m,k} = \texttt{MLP}^{\text{prototype}}(\boldsymbol{\epsilon}_{m,k})$, where each $\boldsymbol{\epsilon}_{m,k}$ is a learned vector of dimension equal to the block size. The MLP is has 4 layers with hidden dimensions 4 times the block-size and the output dimension set to be equal to the block size.

D.4 Image Tokenizer and Transformer

We use discrete VAE, or simply dVAE, as our image tokenizer. We adopt the same architecture as that used in SLATE (Singh et al., 2022a). Like SLATE, the temperature of Gumbel-Softmax (Jang et al., 2016) is decayed using cosine-annealing from 1.0 to 0.1 in the first 30K training iterations. Like SLATE, the dVAE is trained simultaneously with the other parts of the model. We use a patch size of 4×4, meaning that a 128×128-sized image would be represented as a sequence of 1024 discrete tokens. To learn to decode these tokens auto-regressively, we adopt the same transformer decoder and configurations as SLATE without any architectural or significant hyperparameter changes.

D.5 Training

Similar to SLATE (Singh et al., 2022a), we split the learning rates of the dVAE image tokenizer, the Block-Slot Attention encoder, and the transformer decoder. The exact values of the learning rates and the full list of hyperparameters are provided in Table 2. During training, the learning rate is linearly warmed up to its peak value in the first 30K training iterations and then decayed exponentially with a half-life of 250K iterations.

E Additional Experimental Details

E.1 Visualization of Object Clusters using K-means on Blocks

In this section, we provide a more detailed description of the visualizations of Fig. 4 and Fig. 10. To generate these visualizations, we first take a large collection of images. On these, we apply BSA and extract $\mathcal{S} = \{(\mathbf{s}_i, \mathbf{x}_i^{\text{attn}})\}$: a collection of slots $\mathbf{s}_i$ paired with its corresponding object image $\mathbf{x}_i^{\text{attn}}$ of the obtained by multiplying the input image with the attention mask of the slot $\mathbf{s}_i$. From this collection, we take m-th blocks as $\mathcal{S}_m = \{(\mathbf{s}_{i,m}, \mathbf{x}_i^{\text{attn}})\}$ for each block index m. We then apply K-means on the block vectors in $\mathcal{S}_m$ to obtain clusters of blocks. We use $K = 12$. In Fig. 4 and Fig. 10, we visualize the blocks inside some of the clusters by showing the object that those blocks are representing. In the top-left of Fig. 4, we show two of the clusters for the block index 6 for CLEVR-Tex by showing the object images of blocks that lie in those clusters. It shows that all the sphere objects had a representation of block 6 close to each other. Similarly, all the cylinder objects also have block 6 representations that are close to each other. This shows which concept each block is capturing and which it is abstracting away. That is, in this case, block 6 is capturing the shape property of the object but abstracting away other properties such as material or position.

E.2 Selecting Blocks for Swapping

We selected the blocks representing a specific property by first applying K-means on each m-th blocks as in the qualitative analysis of Fig. 4. We then manually inspected the resulting object clusters, and identified, for each block, which property a block is specializing in.

F Limitations and Future Work

The proposed model also shows some limitations. First, although our model significantly outperforms the baselines, the absolute performance is still far from being perfect. Second, the vector-formed factors in our model make slots larger and thus take more computation. Lastly, our transformer decoder can be computationally more expensive than the mixture decoders.

A few future directions are as follows. First, it would be interesting to extend the model to make it work on more complex natural scenes even though the current dataset is already much more complex than those in a similar line of research (Greff et al., 2019; Locatello et al., 2020). Second, it would be interesting to study the multi-modal aspect of the proposed model, particularly with language. Finally, extension to video learning is also interesting.